Generating Explanations in Context

GENERATING EXPLANATIONS IN CONTEXT

Giuseppe Carenini
Department of Computer Science
University of Pittsburgh
Pittsburgh, PA 15260
U.S.A.
Telephone: (412) 624-9185
carenini@cs.pitt.edu

Johanna D. Moore
Department of Computer Science
and
Learning Research and Development Center
University of Pittsburgh
Pittsburgh, PA 15260
U.S.A.
Telephone: (412) 624-7050
FAX: (412) 624-9149
jmoore@cs.pitt.edu

Keywords: Explanation, Dialogue Management, Tutoring Systems, Natural Language Generation

Abstract

If user interfaces are to reap the benefits of natural language interaction, they must be endowed with the properties that make human natural language interaction so effective. Human-human explanation is an inherently incremental and interactive process. New information must be highlighted and related to what has already been presented. In this paper, we describe the explanation component of a medical information-giving system. We describe the architectural features that enable this component to generate subsequent explanations that take into account the context created by its prior utterances.

GENERATING EXPLANATIONS IN CONTEXT

Giuseppe Carenini and Johanna D. Moore

University of Pittsburgh
Department of Computer Science
Pittsburgh, PA 15260
Phone: (412) 624-8408
{carenini, jmoore}@cs.pitt.edu

ABSTRACT

If user interfaces are to reap the benefits of natural language interaction, they must be endowed with the properties that make human natural language interaction so effective. Human-human explanation is an inherently incremental and interactive process. New information must be highlighted and related to what has already been presented. In this paper, we describe the explanation component of a medical information-giving system. We describe the architectural features that enable this component to generate subsequent explanations that take into account the context created by its prior utterances.

KEYWORDS: Explanation, Dialogue Management, Tutoring Systems, Natural Language Processing

INTRODUCTION

When humans engage in dialogue, they freely exploit all aspects of the mutually known context, including the previous discourse. The simplest example of this phenomenon is the use of pronouns to refer to previously invoked objects. Computer-generated discourse that does not draw on the previous discourse in ways like this seems awkward, unnatural, or even incoherent. Strategies for using the discourse history in generating utterances are therefore of great importance for building computer systems that are intended to carry on a dialogue with their users.

The effect of the discourse history is especially important in natural language generation for tutorial and information-giving applications. Effective explanation is essentially incremental and interactive. When the goal is to provide missing information in a way that will facilitate understanding and learning, it is vitally important to form an accurate, evolving model of the knowledge to which the missing information is to be added. New information must be highlighted and related to what the information-seeker already knows. In realistic tutoring applications, any effective explanation will require a detailed model of the previous discourse in order to relate new information effectively to recently conveyed material and to avoid repeating old material that would distract the user from what is new. Since this detailed model will in general involve many assumptions that could be incorrect, it is essential to adopt a style of explanation that will give the user opportunities to provide feedback. If the feedback indicates that the assumptions were incorrect, the explainer will need to refer to the discourse history in making repairs and replanning.

In such applications, lessons that computational linguists have learned in other areas about the importance of context and dialogue history reinforce what educational psychologists have learned about effective instruction.

A SAMPLE INTERACTION

We are involved in a research effort whose goal is to improve the information exchange between physicians and patients in a clinical setting. Initially, we are focusing on migraine patients who require periodic interaction with their physicians for effective management of their condition. The system is intended to provide patients with information about their disease, possible therapies, and

USER	Describe Inderal.	[1]
SYSTEM	Inderal is a drug that is used for prophylactic treatment of migraine. Drugs for prophylactic treatment are intended to reduce or prevent further migraine attacks. These are drugs that you must take every day, whether or not you have a migraine. In general, prophylactic treatment is suitable for patients with frequent migraines. Inderal may not be suitable for you if you have diabetes, problems with your heart, or a severe pulmonary disease such as asthma. Some side effects of Inderal are lowered blood pressure, fatigue, and depression.	[2]
USER	Describe Elavil.	[3]
SYSTEM	Like Inderal, Elavil is used for prophylactic treatment of migraine. Elavil may not be suitable for you if you have irregular heartbeat or if you experience dizziness when you stand up. Elavil is better than Inderal for patients who experience depression, because Elavil is actually an anti-depressant. However, Elavil has other possible side effects including dry mouth, difficulty urinating, and weight gain.	[4]
USER	Describe Cafergot.	[5]
SYSTEM	Cafergot is very different from the drugs we've been talking about. Cafergot is used for abortive migraine treatment, not for prophylactic treatment. Drugs for abortive treatment are intended to stop a headache attack as soon as it starts. You must take an abortive drug as soon as you get a sign that you are going to get a headache. In general, abortive drugs are suitable for patients whose migraines are infrequent. Cafergot may not be suitable for you if you have high blood pressure, heart problems, numbness in an arm or leg when you have a migraine, or if you smoke or are pregnant. A common side effect of Cafergot is nausea. When taking Cafergot, you must be sure not to exceed the recommended dosage, otherwise you may get ergot poisoning.	[6]

Figure 1: Sample Dialogue

medications [3, 15]. We have built a knowledge base that can support explanations of the common therapies, and the side effects of those therapies.

In designing this system, we are making extensive use of empirical data, including ethnographic studies of doctor-patient interactions [6], interviews with patients, and interviews with physicians. The sample dialogue shown in Figure 1 is based upon phenomena we observed in naturally occurring interviews with physicians. This example typifies the type of interaction a patient can have with our system. Patients can ask questions by constructing items from a set of hierarchical menus or by using the mouse to highlight objects on the screen. In this case, the patient first asks the system to describe a drug by constructing the question 'Describe Inderal' (turn 1). The system plans and generates a response for this question in turn 2. The user then asks the system to 'Describe Elavil' (turn 3). Note that although the user asks exactly the same type of question in turn 3 as she did in turn 1, the system's answer in turn 4 is quite different from the answer it generated in turn 2. This is because the answer in turn 4 is affected by the dialogue context that has been created in turns 1–3.

Let us examine response 4 (R4) in detail to see how it is affected by response 2 (R2). The first sentence of R4 points out that Elavil is used for the same type of therapy as Inderal. Pointing out similarities and differences between the object currently being discussed and objects previously discussed is one way in which explanations are affected by prior explanations. Further note that in R4 the system does not explain what prophylactic treatment means because it has done so previously in R2. This illustrates another type of contextual affect; information that has been presented in previous explanations is omitted. In the last 2 sentences of R4, Elavil's contraindications and side effects are contrasted with those of Inderal.

In the remainder of this paper, we describe how a system can produce the type of context-sensitive responses illustrated in this sample dialogue. Our approach to the generation of context-sensitive responses extends previous work by Moore, Paris and Swartout on building an explanation generator for an explainable expert system framework known as EES [16, 17]. We briefly review that explanation architecture, and then describe how we have extended it to allow our system to generate context-sensitive utterances.

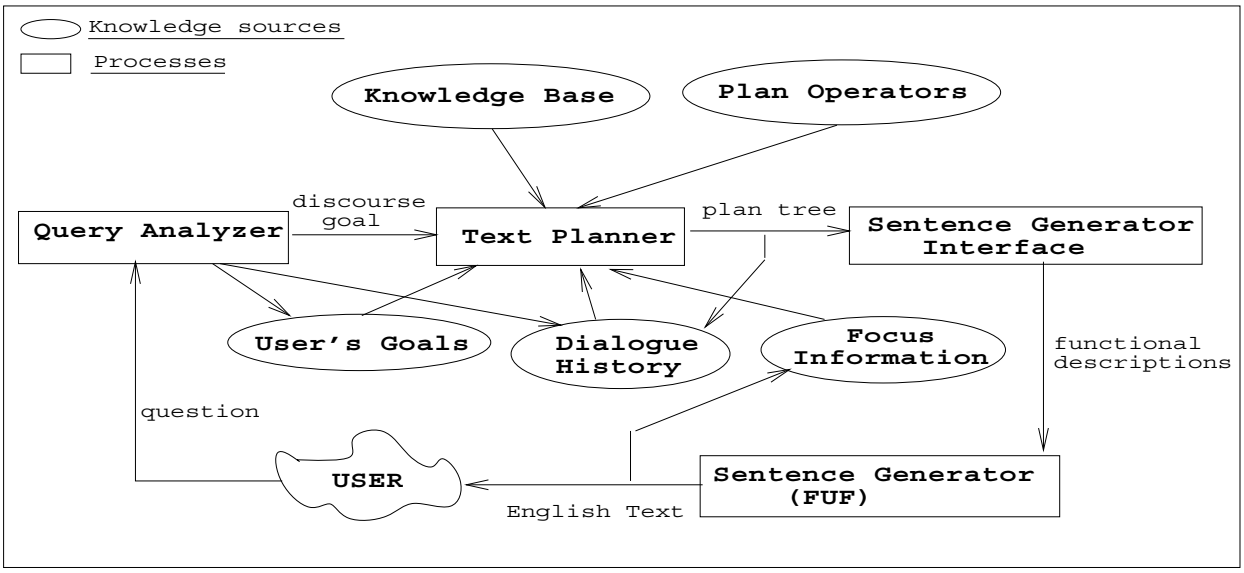

Figure 2: Architecture of Explanation System

OVERVIEW OF THE EES EXPLAINER

Briefly, the EES explanation generator works in the following way. When the user provides input to the system, the query analyzer interprets the question and forms a *communicative goal*. The task of the explanation generator is to synthesize a response to achieve this goal. A communicative goal represents an abstract specification of the answer to be produced, e.g., 'describe an object', 'justify a recommendation'.

The system's knowledge about explanation is contained in a library of plan operators that can be used to achieve the system's communicative goals. In general, there may be many plan operators available to achieve a given goal. In order to choose among the candidate strategies, the planner contains a set of *selection heuristics* that take into account (among other factors) what the user knows (as indicated in the user model), the relative specificity of the operators, and information about whether or not the operators require assumptions to be made.

When a communicative goal is posted, the planning mechanism uses its operators to construct a *text plan* for an explanation. As the system plans an explanation, it records the goal/subgoal structure of the response being produced. In addition, it keeps track of any assumptions it makes about what the user knows, as well as alternative strategies that could have been chosen at any point in the planning process (but were not selected). A completed text plan is an explicit representation of the planning or "design" process that produces an explanation.

The system then presents the explanation to the user, retaining the plan that produced it in a dialogue history. The dialogue history is a record of the conversation that has occurred thus far and includes the user's utterances as well as the text plans that led to the system's responses. After it produces an utterance, the system awaits the user's feedback. If the user asks a follow-up question or indicates that she does not understand the explanation, the system examines the dialogue history to determine how to respond.

EXTENDING THE EXPLAINER

In the EES explainer, the dialogue history is used to determine how to respond to follow-up questions in the following ways. First, the system makes use of the information recorded in a text plan in order to interpret underspecified questions such as "Why?". Because, a text plan records the intentional structure [8] of an explanation as well as information about the order in which individual clauses were presented, it can be used to determine which propositions are currently in the focus of attention, and which were previously in focus. The system interprets "Why?" as a request to justify the proposition currently in focus unless information in the user model indicates that the

user already knows the justification or the previous dialogue history indicates that the justification has previously been given. To find other likely interpretations, the system looks at the next most recently focussed proposition, and so on, until an interpretation is found. For a more detailed discussion, see [17].

Second, if the user asks "Huh?", the EES system reasons about the text plan that produced its previous explanation in order to determine which of its communicative goals have failed or which of the assumptions made may have been erroneous. The system has a set of *recovery heuristics* that guide the planning of elaborating responses [14]. These heuristics tell the system how to proceed depending on the type of goal that was being achieved and the strategy that was to achieve it.

Third, to avoid producing the same answer if a question is asked a second time, the system looks through the dialogue history to determine if the communicative goal corresponding to the question has ever been posted before. If it has, the system notes which strategy was used in the previous case and employs its recovery heuristics to choose an alternative strategy [16].

Here, we have extended the EES explainer by augmenting the ways in which the information recorded in the dialogue history affects each new explanation as it is planned. The general idea is that the previous dialogue should influence the answer to *every* subsequent question, not just explicit follow-up questions such as "Why?" and "Huh?"

An overview of the extended explanation architecture is shown in Figure 2. In order to exploit previous explanations in this more general way, our system makes use of three sources of knowledge:

- the dialogue history which includes the user's utterances and the text plans that led to the system's responses
- information about the the user's goals, as inferred from user's utterances/questions
- a focus history, indicating the objects and relations that are in focus at each dialogue turn.

The text planner utilizes these knowledge sources via two mechanisms. First, we have augmented the plan operator library with operators that implement context-sensitive explanation strategies. These plan operators explicitly check aspects of the context that are modeled in the three knowledge sources discussed above, and produce explanations that are sensitive to various contextual features. Second, we have identified general heuristics that are applicable to all communicative goals. For example, the planner need not expand a communicative goal if it has already been achieved in a previous text plan.[1]

A DETAILED EXAMPLE

We now consider how our system can produce the behavior illustrated in the sample dialogue in Figure 1. When the user asks the system to 'Describe Inderal' in turn 1, the query analyzer interprets this question and posts the goal `(KNOW-ABOUT H INDERAL)`. The planner searches its operator library to find an operator capable of achieving this goal, and finds Operator 1 (Op1) shown in Figure 3. This operator encodes a schema-like [12] strategy for describing a drug. This strategy was derived from our analysis of transcripts of doctor-patient interactions and interviews conducted with physicians.

To determine whether this operator can be used in the current situation, the planner checks its constraints. If a constraint predicate includes only bound variables, then the planner verifies the constraint against the knowledge base. For example, the first constraint in Op1 `(ISA ?d DRUG)` checks the domain knowledge to verify that `INDERAL` is of type `DRUG`. Alternatively, if a constraint predicate contains variables that are not yet bound, the planner searches the system's knowledge bases for acceptable bindings for such variables. For example, to check the constraint `(USE ?d ?t)` where `?d` is bound to `INDERAL`, but `?t` is not bound, the planner searches the medical knowledge base and finds that the variable `?t` can be bound to `PROPHYLACTIC-MIGRAINE-TREATMENT`. Finally, constraints that include the special form `SETF` simply set a variable to the value indicated. Therefore, all the constraints on Op1 are verified, and the operator is chosen.

Next, to expand the operator, the planner posts the subgoal appearing in the nucleus[2]

[1] The knowledge encoded in the general heuristics *could* be encoded as constraints on individual operators, but this would lead to a combinatorial explosion in the size of the operator space.

[2] The terms *nucleus* and *satellite* come from Rhetorical Structure Theory (RST). For more details about RST, see [10].

Operator1

```
EFFECT: (KNOW-ABOUT H ?d)
CONSTRAINTS: (AND (ISA ?d Drug)
                  (Use ?d ?t)
                  (SETF ?attribute (Other-Use
                                    Contraindication
                                    Side-Effect
                                    Warning))
NUCLEUS: (KNOW-ABOUT H (Use ?d ?t))
SATELLITES: (FOR-ALL ?attribute (KNOW-ABOUT H (?attribute ?d)))
```

Operator2

```
EFFECT: (KNOW-ABOUT H (?r ?arg1))
CONSTRAINTS: (?r ?arg1 ?x)
NUCLEUS: (KNOW-ABOUT H (?r ?arg1 (SET ?x)))
```

Operator3

```
EFFECT: (KNOW-ABOUT H (?r ?arg1 ?arg2))
CONSTRAINTS: (IN-DIALOGUE-HISTORY (KNOW-ABOUT H (?r ?x ?arg2)))
NUCLEUS: (KNOW-ABOUT H (SAME-AS (?r ?x ?arg2)
                                (?r ?arg1 ?arg2)))

"Like Inderal, Elavil is used for prophylatic treatment of migraine.."
```

Operator4

```
EFFECT: (KNOW-ABOUT H (Other-Use ?arg1 ?arg2))
CONSTRAINTS: (IN-DIALOGUE-HISTORY (KNOW-ABOUT ?H (Side-Effect ?x ?arg2)))
NUCLEUS: (INFORM H (IS-BETTER ?arg1 ?x ?p
                              (RES (patient ?p)
                                   (experience ?p ?arg2))))
SATELLITES:(EVIDENCE (Other-Use ?arg1 ?arg2))

 "Elavil is better then Inderal for patients who experience depression,
 because Elavil is actually an anti-depressant..."
```

Operator5

```
EFFECT: (KNOW-ABOUT H (Use ?d ?t1))
CONSTRAINTS: (AND (CURRENT-GOAL H ?t2)
                  (NOT (EQUAL ?t1 ?t2))
                  (Used-For ?t2 ?m)
                  (IN-DIALOGUE-HISTORY-CONSECUTIVE (KNOW-ABOUT H ?m)))
NUCLEUS: (CONTRAST-CASE ?d (SET ?m) (Use))
SATELLITES: (BACKGROUND (DIFF ?d (SET ?m)) Use)

"Cafergot is very different from the drugs we've been talking about.
Cafergot is used for abortive treatment, not for prophylactic treatment...."
```

Operator6

```
EFFECT: (KNOW-ABOUT H ?p)
CONSTRAINTS: NIL
NUCLEUS: (INFORM H ?p)
```

Figure 3: Sample Operators

field of the operator, `(KNOW-ABOUT H (USE INDERAL PROPHYLACTIC-MIGRAINE-TREATMENT))`, and then the subgoals appearing in the satellite. The `FORALL` special form causes a subgoal to be posted for each of the bindings of the variable named by its first argument. Expanding the satellites of Op1 will therefore cause four additional subgoals to be posted.

The planner must then find operators for achieving each of the subgoals. To achieve the first subgoal, `(KNOW-ABOUT H (USE INDERAL PROPHYLACTIC-MIGRAINE-TREATMENT))`, the planner uses Op6 which encodes the simple strategy: to make the hearer know any proposition `?p`, simply inform her of `?p`. Speech acts, e.g., `INFORM` and `RECOMMEND`, are the primitives of our text planning system. When a subgoal has been refined to a speech act, the *sentence generator interface* constructs a *functional description (FD)* for the speech act. When text planning is complete, these FDs are passed to the FUF sentence generator [5] which produces the actual English text, see Figure 2.

In the process of building an FD, new text planning goals may be posted as side effects. This occurs because the text planner reasons about concepts and processes in the system's knowledge representation language. Only when building FDs does it consider how these concepts will be realized in text. To provide an informative and understandable text, the system must phrase its utterances using terms the user knows and understands. To do this, as the system is building the FD, it checks the user model to see if each term that will be mentioned in the text is known to the user. In transforming `(INFORM H (USE INDERAL PROPHYLACTIC-MIGRAINE-TREATMENT))`, the interface notes that the user does not already know the con-

cept PROPHYLACTIC-MIGRAINE-TREATMENT, therefore it posts a subgoal to describe this term, as shown in the system's utterance in turn 2 of the sample dialogue. The remainder of the explanation in turn 2 results from expanding the satellite subgoals in a similar manner.

The user then asks the system to 'Describe Elavil' in turn 3. Op1 is again chosen to achieve the goal (KNOW-ABOUT H ELAVIL). However, this time the planner finds two applicable operators for achieving the subgoal (INFORM H (USE ELAVIL PROPHYLACTIC-MIGRAINE-TREATMENT)), namely Op3 and Op6. Note that the constraint of Op3 (IN-DIALOGUE-HISTORY (KNOW-ABOUT H (?r ?x ?arg2))) (where ?r is bound to USE and ?arg2 to PROPHYLACTIC-MIGRAINE-TREATMENT) is satisfied by binding ?x to INDERAL because the system achieved the goal (KNOW-ABOUT H (USE INDERAL PROPHYLACTIC-MIGRAINE-TREATMENT))) in its previous explanation.

The system has a selection heuristic that guides it to choose the most specific operator. Other things being equal, the operator with the most constraints is the most specific.[3] Refining this operator leads the system to generate the text "Like Inderal, Elavil is used for ...". A similar situation occurs when expanding the subgoal (KNOW-ABOUT H (OTHER-USE ELAVIL DEPRESSION)). Here Op2 and Op4 are candidates, and Op4 is chosen because it is more specific. The constraint on Op4 is satisfied because a SIDE-EFFECT of Inderal is actually an OTHER-USE of Elavil. The text that Op4 generates in this case is shown under its definition in Figure 3.

Thus we see that, by checking for the existence of certain subgoals in the dialogue history, Op3 and Op4 enable the system to generate the context-sensitive response in turn 4. Because they are chosen over the less specific operators Op6 and Op2, the system describes Elavil differently than it described Inderal.

In addition, note that the system did not explain the term PROPHYLACTIC-MIGRAINE-TREATMENT when describing Elavil. This is because when the system is planning utterance 4 and attempting to determine whether this term is known to the user, it finds that the term was explained in the previous text (i.e., the goal (KNOW-ABOUT H PROPHYLACTIC-MIGRAINE-TREATMENT) appears in a previous text plan), and therefore it does not re-explain this term.

Finally, the user asks the system to 'Describe Cafergot' (turn 5). In this case, after Op1 is expanded, the system chooses Op5 instead of Op6 to achieve the goal (KNOW-ABOUT H (USE CAFERGOT ABORTIVE-MIGRAINE-TREATMENT)). The first constraint on this operator checks the system's model of the user's goals. This model is built by inferring the user's goals from her questions. In the current system, we have implemented a simple version of a standard plan recognition technique [1]. When the interaction begins, the system assumes that the user's goal is to find out information about possible treatments for migraine. When the user asks about a drug, the system assumes that the user wishes to take this drug to treat her migraines and also to follow the type of therapy this drug is suitable for (e.g., prophylactic or abortive.) So, for example, when the user asks the system to 'Describe Inderal', the system attributes to the user the subtree of goals that is circled in Figure 4. When the user asks about Cafergot, the system infers that her current goal has switched from considering a prophylactic treatment to considering an abortive treatment. The remaining constraints on Op5 recognize this situation and allow the system to generate the explanation in turn 6, which differs from both of the previously generated explanations.

It is worth noting that while Op1 and Op4 are clearly domain dependent, since OTHER-USE and SIDE-EFFECT are attributes of the concept DRUG in our knowledge base; the remaining operators are domain independent. Op2 and Op3 are applicable to any relation, whereas Op5 is applicable in any system where the abstract notions of 'goal' and 'use of a resource for achieving a goal' are explicitly represented. Finally Op6 is very general and will be applicable in any system.

RELATED WORK

Computational linguists have investigated how the context provided by the previous discourse should affect the generation of referring expressions, including pronominalization decisions. For instance, see [11, pp. 218–220],[4]. Granville classifies the relations between a referent and its last point of mention and provides a set of rules for making subsequent references to that item [7].

Others investigated how a more extensive discourse history could affect other aspects of the

[3] See [13] for details.

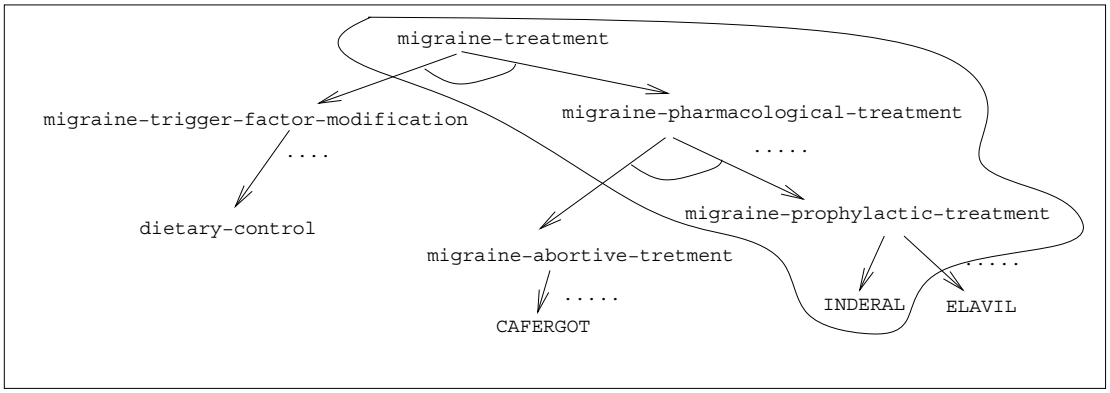

Figure 4: System's Model of User's Goals

response. For example, XPLAIN [18] is a medical expert system capable of justifying its recommendations regarding the dosage of the drug Digitalis. By keeping track of the last few justifications produced and referring to an explicit representation of causal knowledge in the domain, XPLAIN is able to suggest simple analogies with previous explanations and omit portions of a causal chain that have been presented in an earlier explanation. However, this is the only type of contextual effect implemented in XPLAIN, and it was done so using an ad hoc technique to provide this one effect. We are attempting to provide a more general approach.

McKeown carried out a preliminary analysis of how previous discourse might affect a system's response to users' requests to describe an object or compare two objects. McKeown's analysis considered how much and what information should be kept in the dialogue history [12]. She found that by simply maintaining a list of the questions that had been asked, it was possible to avoid certain types of repetition. For example, when asked to compare two objects, if one of the objects has been previously defined, information about that entity should not be included in the current comparison. She further found that if the system were to keep track of the exact information that was provided previously, it could create a text that contrasts or parallels an earlier one.

While McKeown's analysis was fairly detailed, no dialogue history was maintained in the implementation and none of the suggestions for how responses could be altered if such a history existed were actually implemented or tested. We have devised a way for explanation strategies to make use of the information stored in a dialogue history, and have implemented these strategies.

FUTURE WORK

Many issues we have presented in the paper deserve future work. We are currently working in collaboration with the other members of our group in order to determine the types of questions the system should answer, as well as the goals that the user may want to achieve by using the system. This will enable us to understand the goals and plans behind the other types of questions the user asks.

We are also interested in investigating how far back in the dialogue history we should allow the system to search when it is attempting to satisfy operator constraints. This issue requires experimental studies with real users. We plan to evaluate how the effectiveness of the explanations produced by the system changes as we vary the upper bound on the depth of this search.

Finally, we are considering alternative ways to modify a current explanation plan because of the effects of prior explanations. In the current scheme, we have created additional operators whose constraints check for certain patterns in the user model and dialogue history. This may lead to a proliferation of plan operators. An alternative approach is to apply techniques from the work in plan reuse and adaptation that have emerged in the the Artificial Intelligence planning community (e.g., [2, 9]), and we are considering this alternative.

ACKNOWLEDGEMENTS

The research described in this paper was supported by Grant No. R01 LM05299-01 from the National Library of Medicine, National Institute of Health. Johanna Moore is also supported by a Research Initiation Award from the National Science Foundation.

We would like to thank Dimitra Keffalonitou for collecting and transcribing interviews with neurologists, and for implementing the first prototype of the work reported here.

REFERENCES

1. Allen, J. F., and Perrault, C. R. Analyzing intention in utterances. *Artificial Intelligence 15* (1980), 143–178.

2. Alterman, R. An adaptive planner. In *Proceedings of the Fifth National Conference on Artificial Intelligence* (Philadelphia, Pennsylvania, August 11-15, 1986), pp. 65–69.

3. Buchanan, B. G., Moore, J., Forsythe, D., Banks, G., and Ohlsson, S. Involving patients in health care: Using medical informatics for explanation in the clinical setting. In *Proceedings of the Symposium on Computer Applications in Medical Care* (Washington, D. C., 1992). *To appear.*

4. Dale, R. Cooking up referring expressions. In *Proceedings of the Twenty-Seventh Annual Meeting of the Association for Computational Linguistics* (Vancouver, B.C., Canada, June 26-29 1989), pp. 68–75.

5. Elhadad, M. FUF: the universal unifier user manual version 5.0, October 1991.

6. Forsythe, D. E. Using ethnography to build a working system: Re-thinking our basic design assumptions. In *Proceedings of the Symposium on Computer Applications in Medical Care* (Washington, D. C., 1992). *To appear.*

7. Granville, R. Controlling lexical substitution in computer text generation. In *Proceedings of the Tenth International Conference on Computational Linguistics* (Stanford University, July 1984), pp. 381–384.

8. Grosz, B. J., and Sidner, C. L. Attention, intention, and the structure of discourse. *Computational Linguistics 12*, 3 (1986), 175–204.

9. Kambhampati, S., and Hendler, J. A. A validation structure based theory of plan modification and reuse. *Artificial Intelligence* (in press).

10. Mann, W. C., and Thompson, S. A. Rhetorical Structure Theory: Towards a functional theory of text organization. *TEXT 8*, 3 (1988), 243–281.

11. McDonald, D. D. *Natural Language Production as a Process of Decision Making Under Constraint.* PhD thesis, Department of Computer Science and Electrical Engineering, Massachusetts Institute of Technology, 1980.

12. McKeown, K. R. *Text Generation: Using Discourse Strategies and Focus Constraints to Generate Natural Language Text.* Cambridge University Press, Cambridge, England, 1985.

13. Moore, J. D. *A Reactive Approach to Explanation in Expert and Advice-Giving Systems.* PhD thesis, University of California, Los Angeles, 1989.

14. Moore, J. D. Responding to "huh?": Answering vaguely articulated follow-up questions. In *Proceedings of the Conference on Human Factors in Computing Systems* (Austin, Texas, 1989), pp. 91–96.

15. Moore, J. D., and Ohlsson, S. Educating patients through on-line generation of medical explanations. In *Proceedings of the Fourteenth Annual Conference of the Cognitive Science Society* (Bloomington, Indiana, August 1988).

16. Moore, J. D., and Paris, C. L. Planning text for advisory dialogues. In *Proceedings of the Twenty-Seventh Annual Meeting of the Association for Computational Linguistics* (Vancouver, B.C., Canada, June 26-29 1989), pp. 203–211.

17. Moore, J. D., and Swartout, W. R. A reactive approach to explanation. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence* (Detroit, MI, 1989), pp. 1504–1510.

18. Swartout, W. R. XPLAIN: A system for creating and explaining expert consulting systems. *Artificial Intelligence 21*, 3 (September 1983), 285–325.